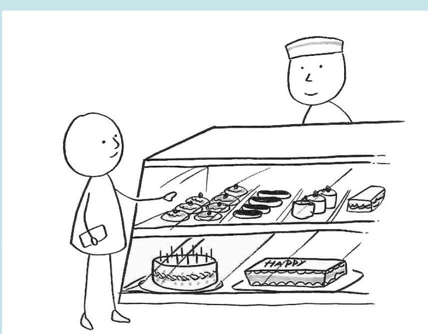
Patisserie: Support for Parameter Sweeps in a Fault-Tolerant, Massively Parallel, Peer-to-Peer Simulation Environment

Tim Schoenharl, Scott Christley
Dept. of Computer Science
and Engineering
University of Notre Dame



Challenge: Parameter Sweeps

- Parameter sweep for large/long running simulations
  - Need to explore parameter space
  - Size of the space
- Highly parallelizable
- Drawbacks of manual/brute force approach:
  - Large space, need direction
  - No idea of shape of parameter space
- Can we perform informed search of the parameter space in an automated way?

Current Challenge: Agent-Based Neural Network Simulation

- Agent-Based Simulation concerned with the effect of local rules on network topology
- Researchers want to conduct a parameter sweep
  - 11 Major Dimensions
  - 1 Run ~ 20-30 minutes

Problem Statement

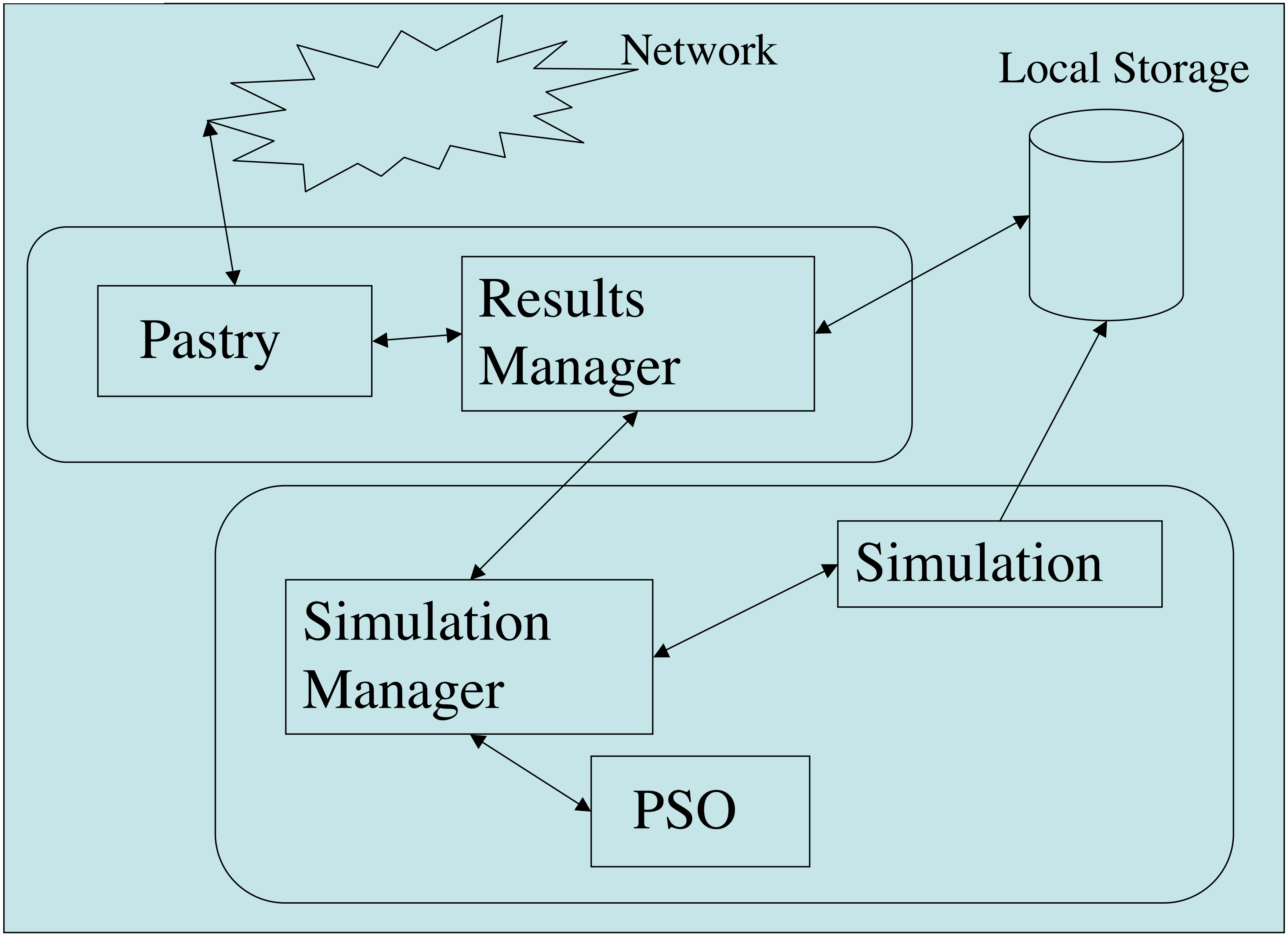
- How can we distribute a simulation across many nodes while maintaining scalability and fault tolerance?
- How can we use loosely-coupled communicating process to perform informed search of the parameter space?
- How can we provide management of simulation results without a central database or shared file system?

Existing Solutions

- AppLeS Parameter Sweep Template [Casanova et al. 2000]
  - Middleware solution using scheduling heuristics and monitors resources.
- Organic Grid [Chakravarti et al. 2004]
  - Tree structured overlay network
- Self-organizing MC [Saramaki 2004]
  - Form overlay network with small-world property

Solution: Patisserie

- Uses the Pastry peer-to-peer framework as a distributed simulation backbone
- Steers ensembles of simulations using the Particle Swarm Optimization algorithm
- Provide peer-to-peer Query API for management of simulation results


Network
Local Storage
Pastry
Results Manager
Simulation
Simulation Manager
PSO

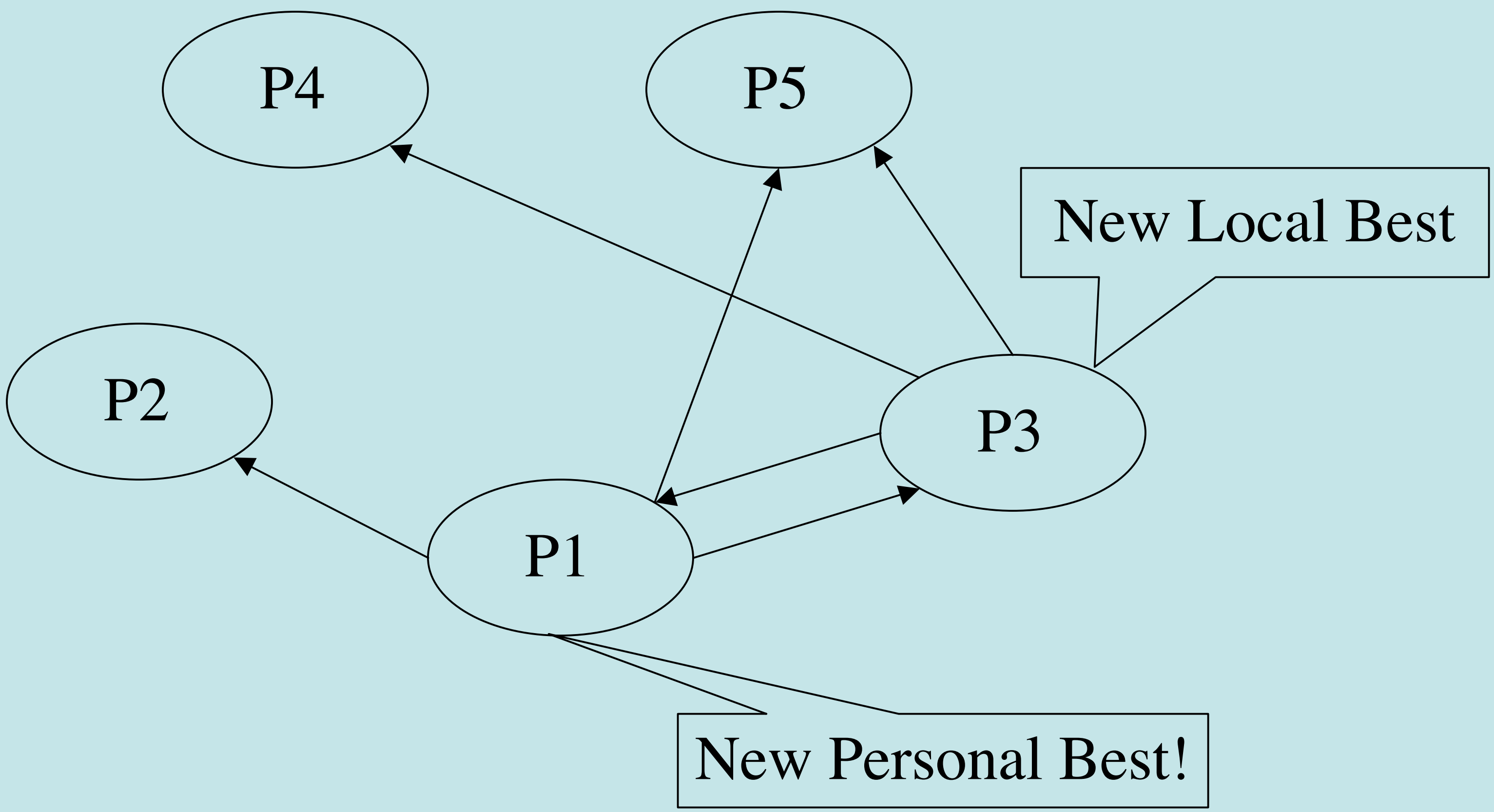
Simulation Steering: Particle Swarm Optimization

- Eberhart and Kennedy 1996
- Each “particle” is an agent with
  - Vector of parameters - current “location”
  - Record of best location
- Method for creating new vector
- Relies on evaluation function
- Particles communicate with their neighbors

PSO Explained

- Each particle maintains
  - Current position and velocity
  - Personal and local best positions
- Particle evaluates current location
- Iteration to next position involves querying neighbors, consulting best position values

PSO Explained



PSO Explained - Optimization

- Particles store location and velocity vectors
- Particles use current location, best location and velocity to choose next location

$$v_{id}(t) = v_{id}(t-1) + \varphi_1(p_{id} - x_{id}(t-1)) + \varphi_2(p_{gd} - x_{id}(t-1))$$

Velocity Update Function

$$x_{id}(t) = x_{id}(t-1) + v_{id}(t)$$

Position Update Function

Simulation Results Management

- Storage of result
  - STORE message type
  - Keep raw data on local node, store referential pointer
- Mapping to key space
  - Pastry defines 160 bit key space
  - Given range of values, scale to the key space
  - Issue: distribution of result values determines distribution of
- Replication of results
  - Replication of raw data files
  - Replication of reference pointers

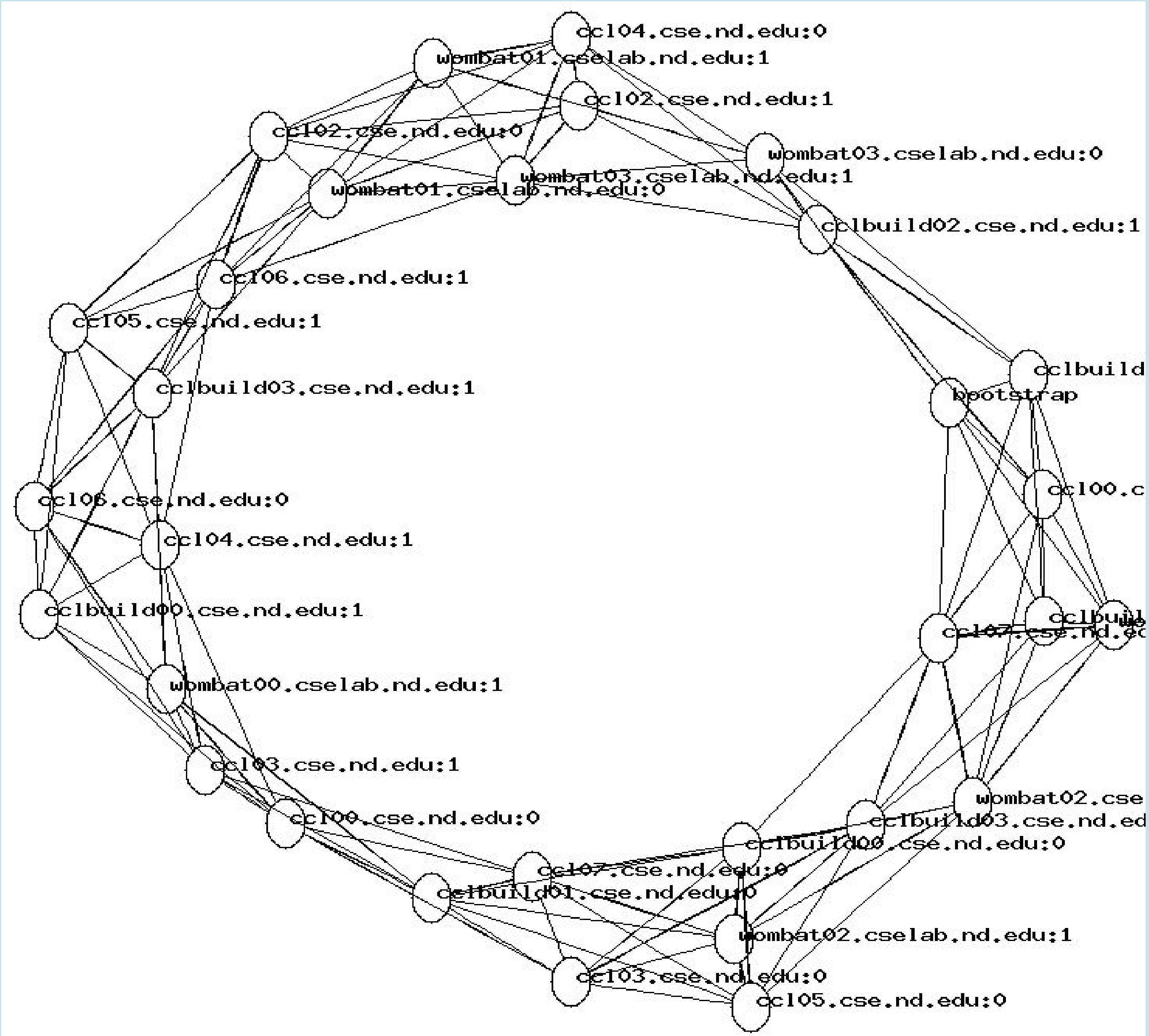
User Queries

- Given a range, return all results in range
- Each node is point in key space and is responsible for keys from prior node key to itself.
- Send Algorithm:
  - Convert lower bound of range to key space
  - Pass query to that node
- Receive Algorithm:
  - Collect all results within range and send back to source node.
  - If upper bound of range > node's key
    - Make node's key the new lower bound
    - Perform send algorithm

Experimental Setup

- Patisserie deployed on the CCL Condor cluster
- Maximum of 16 machines at a time
- Testing Constraints:
  - Virtual Nodes
  - Optimization Functions

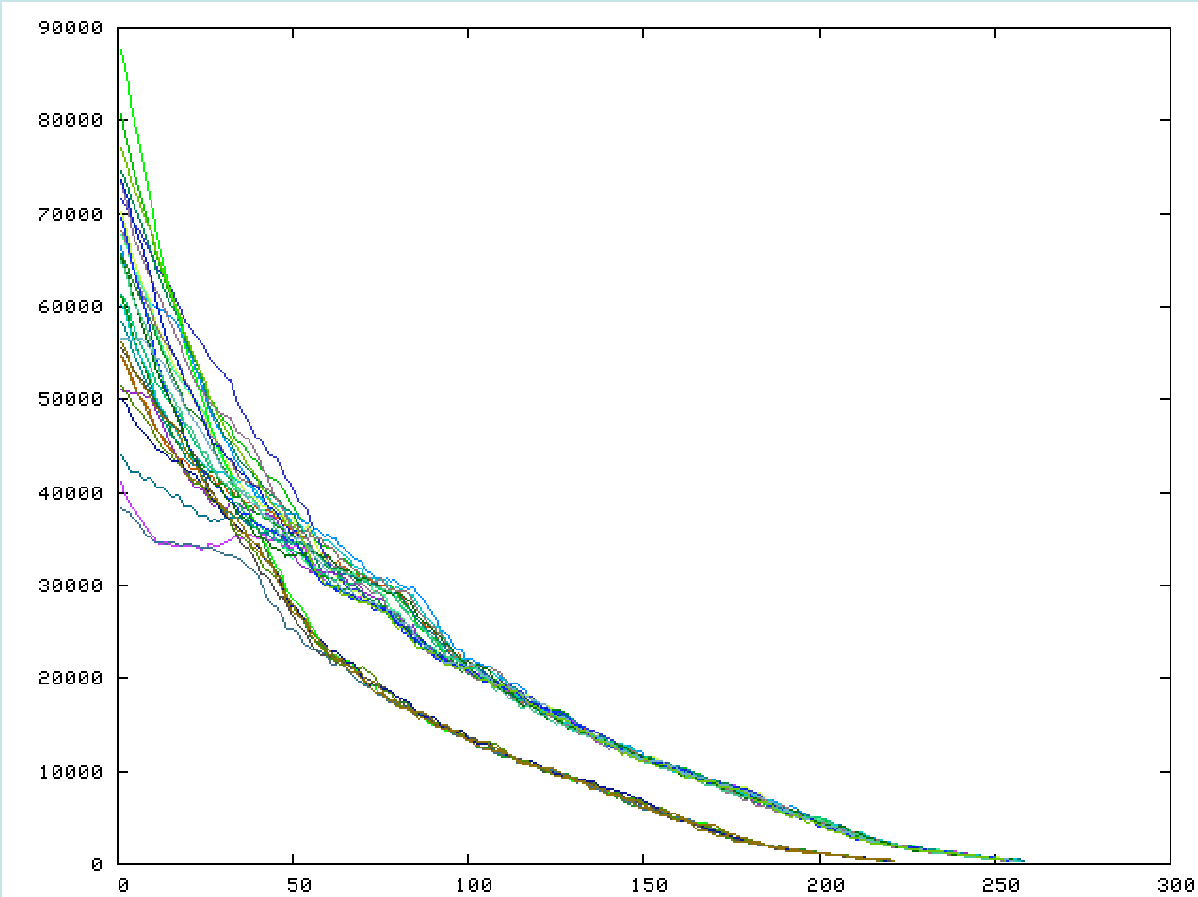
UNIVERSITY OF NOTRE DAME



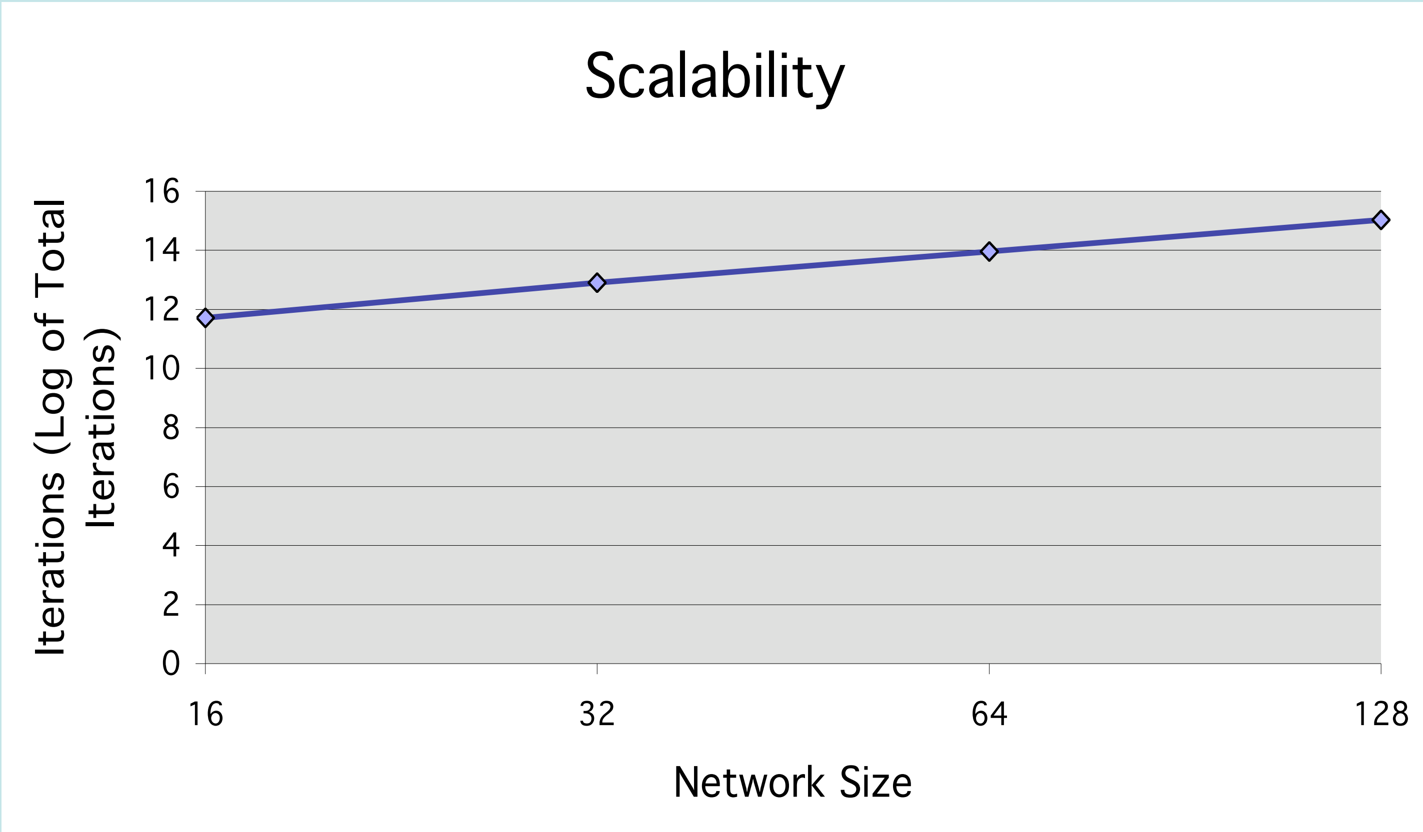
Optimization Functions

- Canonical Examples
  - De Jong's Sphere $\sum_{i=1}^{n} x_i^2$
  - Rastrigin Function $\sum_{i=1}^{n} x_i^2 - 10\cos(2\pi x_i) + 10$
- Optimized DeJong's Sphere to threshold

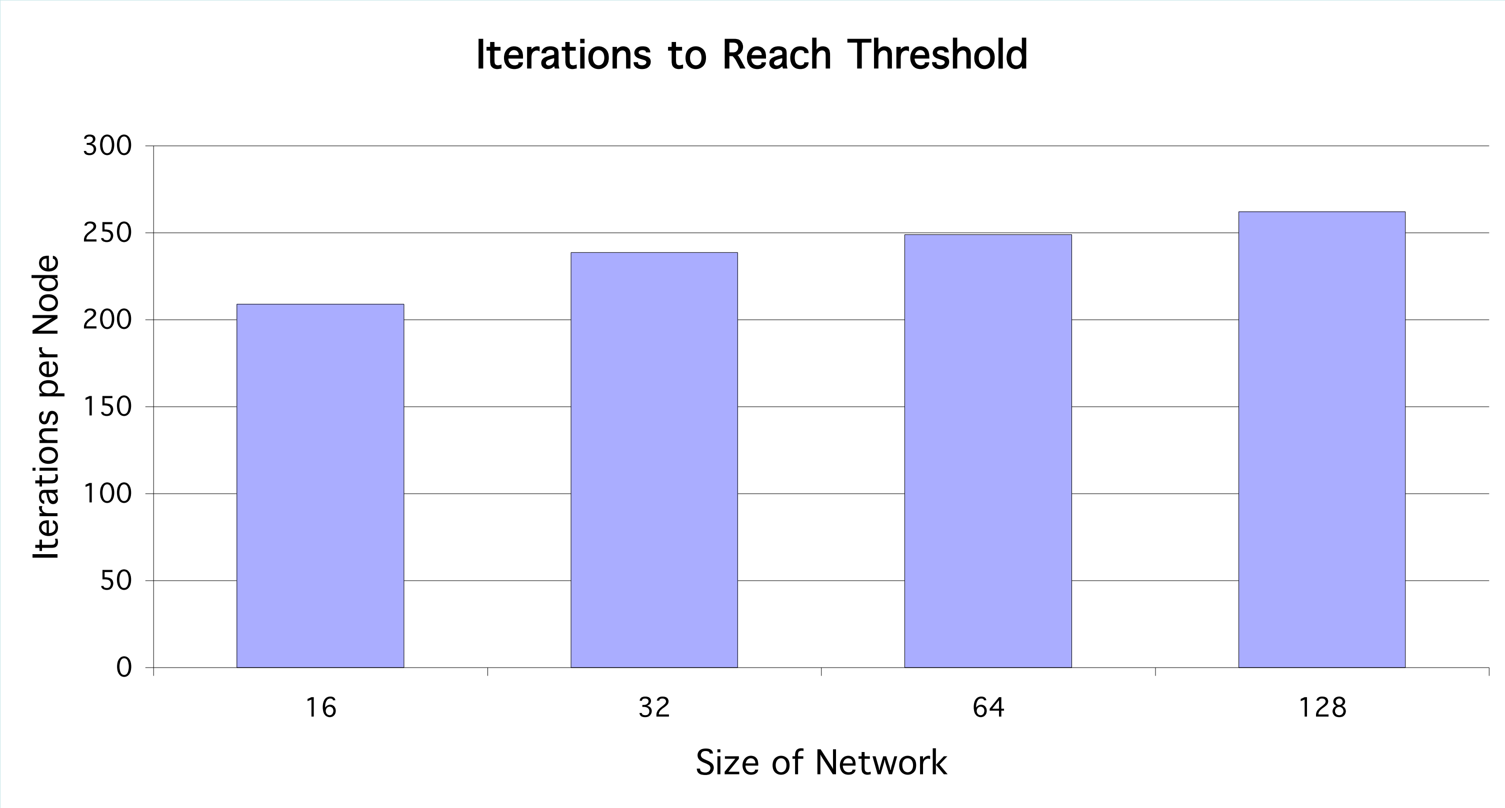
Optimization Behavior



Results


Scalability
Iterations (Log of Total Iterations)
16
14
12
10
8
6
4
2
0
16
32
64
128
Network Size

Results


Iterations to Reach Threshold
300
250
200
150
100
50
0
Iterations per Node
16
32
64
128
Size of Network

Conclusions

- Increasing nodes gives better coverage of parameter space
- Adding more nodes does not provide speedup in optimization
- Distribution of results highly dependent on mapping to key space

Conclusions

- Patisserie provides scalability for distribution of simulations
- PSO takes advantage of peer-to-peer network to provide acceptable optimization behavior
- Simulation results distributed across all nodes, utilizing local storage

Future Work

- Using Pastry for both centralized and peer-to-peer functionality
- Evaluation of Fault-tolerance
- Evaluation of Query performance